COS 497: Computer Science Capstone 2

Sudarshan S. Chawathe

University of Maine

Spring 2013

This course is the third of a three-course sequence designed to guide students in completing the Capstone project in either an independent study, group project, or field experience format. The focus is on the later stages of project work, including completing the programming tasks, evaluating the implemented systems, documenting all work in a project report, demonstrating the work in action, and making a public oral presentation.

News and Reminders:

- Please read the class newsgroup for timely announcements: `umaine.cs.capstone` on NNTP server `news.cs.umaine.edu`. Web interface to get started: `http://cs.umaine.edu/~chaw/news/`.
- The most recent version of this document may be found at `http://cs.umaine.edu/~chaw/cap2/`.
- Please use the PDF version of this document for printing and reference: `cap2.pdf`

Goals and Learning Objectives

Goals

- Integrate knowledge and skills acquired in other courses.
- Learn new material with attention to the learning process.
- Develop the ability to independently explore a topic by discovering, reading, and critiquing prior work.
- Gain experience in contributing to the body of knowledge.
- Gain experience in conducting and documenting experimental studies of programs.
- Improve our programming skills, with attention to software engineering principles.
- Improve our communication skills, with particular emphasis on written communication and, further, well-written programs.
- Practice the appropriate and ethical use of existing material of different kinds, such as source code, services, and documentation.
- Learn how to manage a self-directed project.

Learning Objectives

Students should be able to

- Develop effective learning strategies for continuing acquisition of knowledge and skills.
- Make effective use of the research literature.
- Determine how available software may be used, subject to both common professional standards and the legal licenses governing the software.
- Understand and follow formal and informal standards of the field.
- Choose an appropriate method for contributing their own work (code, documentation, reports) to the profession, including licenses and copyrights that best suit their needs.
- Write code that can be easily used by their peers and others.
- Perform scientifically sound experimental evaluations of their work.

- Evaluate appropriate software engineering methods for individual and team work.
- Present their work in a public forum to their peers and others.

Prerequisites

The three prerequisites for this course are COS 397, senior standing, and permission of the instructor. *Permission to register will be granted only to those students who have made enough progress in their project work to indicate a high likelihood of timely project completion.* A key factor is the recommendation from the project advisor with additional input from the academic advisor. Students should discuss these prerequisites with their *academic advisors* before seeking help elsewhere.

Contact Information

Class meetings:

Time: Tuesdays & Thursdays, 3:30–4:45 p.m.
Location: Neville Hall, Room 120.

Instructor: Sudarshan S. Chawathe

Office: Neville Hall, Room 224.
Office hours: (Please check for changes.)
Tuesdays & Thursdays: 10:30–11:00 a.m.; 1:45–2:00pm; 3:15–3:30 p.m.; 4:45–5:00 p.m.
Phone: +1-207-581-3930.
Please avoid calling except for truly urgent matters.
Email: `chaw@cs.umaine.edu`
Use email only for messages unsuitable for the newsgroup. (See below.) Please use only this email address and put the string *COS497* near the beginning of the Subject header of your messages to me. *All other messages may be ignored.*
Web: `http://cs.umaine.edu/~chaw/`.

Teaching Assistant: Mark A. Plummer

Office: Boardman Hall, Room 127.
Office hours: (Please check for changes.)
Mondays & Wednesdays: 2:10–3:10 p.m.
Email: `mark.a.plummer@maine.edu`

Online Resources

Class Web site: `http://cs.umaine.edu/~chaw/cap2/`
We will use the class Web site for posting assignments, readings, notes, and other material. Please monitor it.

Class Newsgroup: We will use the local USENET newsgroup `umaine.cs.capstone` on the NNTP (net news) server `news.cs.umaine.edu` for electronic discussions. The Web interface at `http://cs.umaine.edu/~chaw/news/` provides convenient access. Some further, more general, information on USENET appears at `http://en.wikipedia.org/wiki/Usenet`. The newsgroup is the primary forum for electronic announcements and discussions, so please monitor it regularly, and post messages there as well. Unless there is a reason for not sharing a question or comment, please *use the newsgroup, not email*, for questions and comments related to this course.

Class mailing list: *Please make sure you are on the class mailing list.* A sign-up sheet is circulated at the first class meeting. If you miss it, please contact me to get on the list. We will use this mailing list

only for urgent messages because all other messages will go on the class newsgroup. I anticipate fewer than a dozen messages on this list over the semester.

Grading Scheme

Grade components: *Students are expected to complete and submit all assigned coursework in good faith; those who fail to do so may earn a failing grade, regardless of overall numerical score.*

component	% grade	
class participation	5	
classroom exercises and journal	10	
homework assignments	15	
project reports (versions 1, 2, & 3)	25	(5 + 10 + 15)
source code and demo (versions 1, 2, & 3)	25	(5 + 10 + 15)
final oral presentation	20	

Class participation: Students are expected to contribute to learning by asking questions and making relevant comments in class and on the class newsgroup. Quality is more important than quantity. Disruptive activity contributes negatively. See policies below.

Classroom exercises and journal: Our work in the classroom will include a number of short group exercises, meant to solidify understanding of the concepts being discussed. One or more such exercises are likely to be part of most class meetings. Students must maintain a journal of their progress through the course and submit the journal periodically (typically on the dates of quizzes and exams) for grading. The journal consists of neatly organized classroom exercises, student observations on their own learning, and other material as announced in class. The exercises and journal will be graded primarily for effort, group work, and other contributions, and less so for simple correctness. Since attendance is not mandatory (cf. policies), some low-scoring exercises will be dropped for each student. Please ask for clarifications if there are concerns about the interaction of this component and the attendance policy.

Homeworks: Homeworks include programming and non-programming ones, often mixed. No collaboration is permitted. Everyone is encouraged to discuss the problems and solution strategies *at a high level*, but the final solution and details must be individual work. If the boundary between permissible and non-permissible interactions is unclear, please ask for clarifications.

Project Reports: The sequence of three project reports serves to systematically document the project. Further details will follow in class. Students are strongly encouraged to continually seek feedback on their working drafts from their project advisors, Capstone instructor, academic advisors, and others.

Source code and demo: Well packaged and documented source code is an important component of the Capstone project. The code will be evaluated on not only how well it functions but also on aspects such as clarity and elegance. The source code does *not* have to be released under any specific license (although a free software license[1] is strongly recommended); however, no legal encumbrances (such as nondiscolsure agreements) will be entertained. All code must be submitted electronically (only) as outlined in the *Submission Instructions* section below.

Final Oral Presentation: Every student must make an oral presentation of his or her work on a date near the end of the semester. The date will be selected to ensure good attendance by department faculty and others, and will be announced in the first few weeks. The presentations are likely to be scheduled during the last week of classes, or finals week. If you have concerns in this regard, you must voice them early in the semester.

[1] such as one compatible with the Debian Free Software Guidelines.

Policies

Due dates: All due dates and times, as announced in class, are strict, to the second. If you believe your work was delayed by truly exceptional circumstances, let me know as soon as those circumstances are known to you and I will try to make a fair allowance. However, *the default is that you get a zero if you don't turn in the work on time*, and fail the class if you don't turn it in at all (cf. Grade Components above).

Attendance: Although I expect students to attend all class meetings, I will not be taking attendance. *If you miss a class meeting, you are responsible for catching up on the lost material, including any important announcements made in class, on your own.* If you have a valid reason for missing a class, let me know early and I will try to help you make up the class. There will be no make-up exams or quizzes. A missed test earns zero credit. If you have a valid reason for missing a test, let me know as early as that reason is known to you and I will make a fair allowance but there will be no make-up tests in any case.

Classroom activities: This course is based on an active learning format, so effective classroom activities are critical to its success. Students are expected to contribute to their own learning and that of their classmates, and to devote 100% of their attention to these activities while in class. On a similar note, all electronic and other distractions (computers, phones, assorted gizmos, etc.) must be completely silenced and put away for the entire duration of the class. (Students who need any such devices for disability accommodations should follow the guidelines outlined below. Others who need any accommodation in this regard due to special circumstances should make advance arrangements with the instructor.) No food or drink is allowed in class, other than water in a spill-proof container. Students who violate these rules or otherwise cause distractions in class will be asked to leave with *no warning*; habitual violators will face disciplinary action.

Office hours: All students are encouraged to make use of both the instructor's and TA's office hours to further their learning, obtain assistance on homework assignments, obtain feedback on their class performance, etc. However, office hours are not to be used as a substitute for attending and participating in class meetings (see above). Similarly, assistance with homework assignments will be limited to what is appropriate based on fairness to all; students are expected to demonstrate substantial effort on the assignment before seeking assistance.

Make-up classes: I may have to reschedule a few classes due to my other professional commitments. I will make every attempt to minimize the number of such occurrences and to reschedule for a time that works for most students. Further, I will make sure no student is penalized by such occurrences.

Academic honesty (standard university wording): Academic dishonesty includes cheating, plagiarism and all forms of misrepresentation in academic work, and is unacceptable at The University of Maine. As stated in the University of Maine's online undergraduate Student Handbook, plagiarism (the submission of another's work without appropriate attribution) and cheating are violations of The University of Maine Student Conduct Code. An instructor who has probable cause or reason to believe a student has cheated may act upon such evidence, and should report the case to the supervising faculty member or the Department Chair for appropriate action.

Disabilities (standard university wording): If you have a disability for which you may be requesting an accommodation, please contact Ann Smith, Director of Disabilities Services, 121 East Annex, 581-2319, as early as possible in the term.

Special circumstances (standard university wording): In the event of an extended disruption of normal classroom activities, the format for this course may be modified to enable its completion within its programmed time frame. In that event, you will be provided an addendum to the syllabus that will supersede this version.

Readings

This list will be revised and annotated as the semester progresses to reflect, in particular, the topics and papers selected based on class discussions.

Assigned

1. Sudarshan S. Chawathe. Capstone project proposals—suggestions for deeper explorations. Department of Computer Science, University of Maine. `http://cs.umaine.edu/`, February 2008.
2. George D. Gopen and Judith A. Swan. The science of scientific writing. *American Scientist*, 78:550–558, November-December 1990.
3. Gordon Harvey. A brief guide to the elements of the academic essay. Harvard College Writing Program. `http://writingprogram.fas.harvard.edu/`, 2009.
4. Ian Parberry. How to present a paper in theoretical Computer Science: A speaker's guide for students. `http://larc.unt.edu/ian/pubs/speaker.pdf`, 2000.
5. Atmel Corporation. AVR220: bubble sort. `http://www.atmel.com/`, May 2002. Atmel 8-bit AVR Microcontroller Application Note.

Other

1. Herb Sutter. Measuring parallel performance: Optimizing a concurrent queue. *Dr. Dobbs Journal*, 34(1):37–44, January 2009.
2. Timothy Furtak, José Nelson Amaral, and Robert Niewiadomski. Using SIMD registers and instructions to enable instruction-level parallelism in sorting algorithms. In *Proceedings of the 19th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 348–357, San Diego, California, 2007.
3. Jon L. Bentley and M. Douglas McIlroy. Engineering a sort function. *Software–Practice and Experience*, 23(11):1249–1265, November 1993.
4. Derrick Coetzee. An efficient implementation of Blum, Floyd, Pratt, Rivest, and Tarjan's worst-case linear selection algorithm. `http://moonflare.com/`, January 2004.
5. Bingsheng He, Ke Yang, Rui Fang, Mian Lu, Naga K. Govindaraju, Qiong Luo, and Pedro V. Sander. Relational joins on graphics processors. In *Proceedings of the 28th ACM International Conference on Management of Data (SIGMOD)*, Vancouver, Canada, June 2008.
6. Naga K. Govindaraju, Jim Gray, Ritesh Kumar, and Dinesh Manocha. GPUTeraSort: High performance graphics coprocessor sorting for large database management. In *Proceedings of the 26th ACM International Conference on Management of Data (SIGMOD)*, Chicago, Illinois, July 2006.
7. Daniel Cederman and Philippas Tsigas. A practical quicksort algorithm for graphics processors. Technical Report 2008-01, Department of Computer Science and Engineering, Chalmers University of Technology and Göteborg University, Göteborg, Sweden, 2008.
8. Sang-Won Lee and Bongki Moon. Design of flash-based DBMS: an in-page logging approach. In *Proceedings of the 27th ACM International Conference on Management of Data (SIGMOD)*, pages 55–66, Beijing, China, June 2007.
9. Gilad Bracha. Generics in the Java programming language. Tutorial. `http://java.sun.com/`, July 2004.
10. Ken Thompson. Reflections on trusting trust. *Communications of the ACM*, 27(8):761–763, August 1984.
11. Mark C. Hamburg. Two tagless variations on the Deutsch-Schorr-Waite algorithm. *Information Processing Letters*, 22:179–183, 1986.
12. Martin E. Hellman. An overview of public-key cryptography. *IEEE Communications Magazine*, 50(5):42–49, May 2002. Originally published in 16(6), November 1978.
13. Jon Bentley and Don Knuth. Programming pearls: Literate programming. *Communications of the ACM*, 29(5):364–369, May 1986.

14. Jon Bentley, Don Knuth, and Doug McIlroy. A literate program. *Communications of the ACM*, 29(6):471–483, June 1986.
15. Paul E. Black. Dictionary of algorithms and data structures. `http://www.nist.gov/dads/`, September 1998.
16. Lloyd Allison. Suffix trees. `http://www.allisons.org/ll/AlgDS/Tree/Suffix/`, 2008.

Exercises, Homeworks, Tests, and Notes

Material will appear here as we move along the semester. It may be useful to refer to the homeworks and tests from the previous session: `http://cs.umaine.edu/~chaw/201209/cap2/`.

- Class Exercises:
  - CE 01: `hwq/ce01.pdf`.
  - CE 02: `hwq/ce02.pdf`.
  - CE 03: `hwq/ce03.pdf`.
  - CE 04: `hwq/ce04.pdf`.
  - CE 05: `hwq/ce05.pdf`.
  - CE 06: `hwq/ce06.pdf`.
  - CE 07: `hwq/ce07.pdf`.
  - CE 08: `hwq/ce08.pdf`.
  - CE 09: `hwq/ce09.pdf`.
  - CE 10: `hwq/ce10.pdf`.
  - CE 11: `hwq/ce11.pdf`.
  - CE 12: `hwq/ce12.pdf`.
  - CE 13: `hwq/ce13.pdf`.
  - CE 14: `hwq/ce14.pdf`.
  - CE 15: `hwq/ce15.pdf`.
  - CE 16: `hwq/ce16.pdf`.
  - CE 17: `hwq/ce17.pdf`.
  - CE 18: `hwq/ce18.pdf`.
  - CE 19: `hwq/ce19.pdf`.

Submission Instructions

All electronic submissions must be made using the upload interface at `http://cs.umaine.edu/~chaw/u/`. *Electronic submissions in all other forms, such as email or physical media, will be discarded and receive no credit.*

Uploaded *files must be named* following this template:

`cos497-hw01-`*`Lastname-Firstname-N`*`.jar`

The substrings `hw01` and `jar` are replaced by others depending on the material being submitted and N is an arbitrary 4-digit number, such as 4231. Multiple submissions, within reason, may be made by selecting different values of N.

If your upload is successful, you will be presented with a confirmation Web page similar to the following sample. You should record the reported MD5 checksum and timestamp.

```
SUCCESS: Please note the following for your records.

Successfully saved cos497-hw01-Aardvark-Alice-1389.jar.
MD5 checksum: 09ee098b83d94c7c046d6b55ebe84ae1
Timestamp: 2013-01-13 13:32:34
```

If you do not see something very similar then your submission is unsuccessful.